Reporte Técnico de Actividades Práctico-Experimentales Nro. 00X

1. Datos de Identificación del Estudiante y la Práctica

Nombre del estudiante(s)	Ricardo Mathias Ochoa Armijos
Asignatura	Teoría de la programación
Ciclo	1 A
Unidad	3 Unidad
Resultado de aprendizaje de la unidad	Desarrolla aplicaciones utilizando el principio de la programación modular y estructuras de datos simples y/o estáticas compuestas, bajo los principios de solidaridad, transparencia, responsabilidad y honestidad.
Práctica Nro.	001
Tipo	Individual
Título de la Práctica	Construcción de funciones y procedimientos en un lenguaje de programación.
Nombre del Docente	Lissette Geoconda López Faicán
Fecha	Jueves 8 de enero del 2026 Jueves 15 de enero del 2026
Horario	10h30 – 13h30
Lugar	Aula física asignada al paralelo.
Tiempo planificado en el Sílabo	6 horas

2. Objetivo(s) de la Práctica

Aplicar los fundamentos de la programación modular mediante la construcción y uso de funciones y procedimientos, para resolver un problema real, garantizando un código estructurado, reutilizable y correctamente documentado.

3. Materiales, Reactivos, Equipos y Herramientas

- IDE de programación: Visual Studio Code u otro entorno compatible.
- Lenguaje de programación: C (según los contenidos de la unidad).

- Computador personal con sistema operativo Windows, Linux o macOS.
- Material de apoyo en el Entorno Virtual de Aprendizaje (EVA).
- Editores de texto (Word, Google Docs u otros) para la elaboración del informe técnico en formato PDF.
- Conexión a internet estable para acceder a recursos digitales y software en línea.
- Aula física asignada al paralelo.

4. Procedimiento / Metodología Ejecutada

El programa solicita las notas, calcula promedios, verifica que estén dentro del rango permitido y finalmente muestra la nota final junto con un resultado claro: aprobado, supletorio o reprobado. La intención es que el código sea simple, ordenado y fácil de comprender.

5. Resultados

Metodología de aprendizaje: aprendizaje basado en problemas.

Inicio

● Presentación del objetivo de la práctica.

● Explicación de la relevancia de aplicar funciones en la codificación de programas.

● Contextualización del problema: Se requiere desarrollar un programa que calcule la nota final de un estudiante, aplicando funciones y procedimientos para cada componente evaluativo: ○ Calcular nota del Aprendizaje en Contacto con el Docente (ACD): solicita nro de actividades, las notas para cada actividad, calcula promedio, y retorna el total ponderado sobre 2.0.

○ Calcular nota del Aprendizaje Práctico Experimental (APE): solicita nro de actividades, las notas para cada actividad, calcula promedio, y retorna el total ponderado sobre 2.5.

○ Calcular nota del Aprendizaje Autónomo (AA): solicita nro de actividades, las notas para cada actividad, calcula promedio, y retorna el total ponderado sobre 2.0.

○ Calcular Evaluación sumativa (ES): solicita la nota de Aprendizaje Basado en Problemas y Portafolio Digital, calcular el ponderado (60% y 40%), y retornar el total ponderado sobre 3.5.

○ Calcular promedio por unidad: ACD + APE + AA + ES (retorna el promedio para una unidad).

○ Calcular el promedio de la asignatura: promedio simple de acuerdo al número de unidades con la escala cualitativa.

- APROBADO: Si la nota final es mayor o igual a 7.
- SUPLETORIO: Si la nota final es mayor o igual a 2.5 y menor a 7.
- REPROBADO: Si la nota final es menor a 2.5.

○ Las notas deben estar en el rango de 0.0 a 10.0

○ Salida de Resultados: Imprimir la nota cuantitativa y cualitativa final del estudiante.

○ Requerimiento del código: El programa debe estar bien estructurado, con comentarios y mensajes descriptivos que faciliten su uso.

CalculoAsignatura > C APE1.C > ...

```c
#include <stdio.h>

float calcularACD();
float calcularAPE();
```

```c
float calcularAPE() {
    int numeroActividades;
    float notaActividad, notaAcumulativa = 0;

    printf("Ingrese el numero de actividades de APE: ");
    scanf("%d", &numeroActividades);

    for (int i = 1; i <= numeroActividades; i++) {
        do {
            printf("Ingrese la nota de la actividad %d: ", i);
            scanf("%f", &notaActividad);
        } while (notaActividad < 0 || notaActividad > 10);

        notaAcumulativa += notaActividad;
    }

    return (notaAcumulativa / numeroActividades) * 0.25;
}

float calcularAA() {
    int numeroActividades;
    float notaActividad, notaAcumulativa = 0;

    printf("Ingrese el numero de actividades de AA: ");
    scanf("%d", &numeroActividades);

    for (int i = 1; i <= numeroActividades; i++) {
        do {
            printf("Ingrese la nota de la actividad %d: ", i);
            scanf("%f", &notaActividad);
        } while (notaActividad < 0 || notaActividad > 10);

        notaAcumulativa += notaActividad;
    }

    return (notaAcumulativa / numeroActividades) * 0.20;
}
```

```c
float calcularES() {
    float notaExamen, notaPortafolio;

    do {
        printf("Ingrese la nota del examen: ");
        scanf("%f", &notaExamen);
    } while (notaExamen < 0 || notaExamen > 10);

    do {
        printf("Ingrese la nota del portafolio: ");
        scanf("%f", &notaPortafolio);
    } while (notaPortafolio < 0 || notaPortafolio > 10);

    return (notaExamen * 0.60 + notaPortafolio * 0.40) * 0.35;
}

int main() {
    int NUMEROUNIDADES = 3;
    float promedioFinal = calcularPromedioFinal(NUMEROUNIDADES);

    printf("\nSu nota final de la asignatura es: %.2f", promedioFinal);

    if (promedioFinal >= 7) {
        printf("\nExcelente");
    } else if (promedioFinal >= 2.5) {
        printf("\nSupletorio");
    } else {
        printf("\nReprobado");
    }

    return 0;
}
```

```
PS C:\Users\USUARIO\Codigos Ricky\Unidad 3> CD CalculoAsignatura
PS C:\Users\USUARIO\Codigos Ricky\Unidad 3\CalculoAsignatura> .\APE1.exe

Unidad 1
Ingrese el numero de actividades de ACD: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de APE: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de AA: 1
Ingrese la nota de la actividad 1: 9
Ingrese la nota del examen: 9
Ingrese la nota del portafolio: 9

Unidad 2
Ingrese el numero de actividades de ACD: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de APE: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de AA: 1
Ingrese la nota de la actividad 1: 9
Ingrese la nota del examen: 9
Ingrese la nota del portafolio: 9

Unidad 3
Ingrese el numero de actividades de ACD: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de APE: 1
Ingrese la nota de la actividad 1: 9
Ingrese el numero de actividades de AA: 1
Ingrese la nota de la actividad 1: 9
Ingrese la nota del examen: 9
Ingrese la nota del portafolio: 9

Su nota final de la asignatura es: 9.00
Excelente
PS C:\Users\USUARIO\Codigos Ricky\Unidad 3\CalculoAsignatura>
```

6. Preguntas de Control

¿Cuál es la diferencia entre una función y un procedimiento?

Una función devuelve un valor usando return.
Un procedimiento no devuelve ningún valor y se declara con void

¿Qué ventajas aporta dividir un programa en funciones (modularidad)?

Hace el programa más ordenado, fácil de entender y más fácil de corregir.
Cada función hace una sola cosa y el código no se repite.

¿Qué se mejoraría del programa si se tuviera que usarlo para varios estudiantes?

Se podría reutilizar el mismo código para todos los estudiantes, evitar escribir el programa varias veces y calcular las notas de muchos alumnos de forma automática.

7. Conclusiones

El uso de funciones y procedimientos en C permite organizar mejor un programa, hacerlo más claro y más fácil de mantener. Al dividir el código en partes pequeñas, se evitan errores, se reutiliza la lógica y el programa puede adaptarse fácilmente para trabajar con varios estudiantes. En conjunto, la modularidad hace que el programa sea más eficiente, ordenado y práctico para su uso real.